TRANSMITTAL OF APPEAL BRIEF (Large Entity)	Docket No. INTL-0038-US

In Re Application Of: **JOHN W. MERRILL**

Serial No.	Filing Date	Examiner	Group Art Unit
09/115,359	July 14, 1998	R. Sax	2743

Invention: **AUTOMATIC SPEECH RECOGNITION**

RECEIVED
FEB 26 2001
Technology Center 2600

TO THE ASSISTANT COMMISSIONER FOR PATENTS:

Transmitted herewith in triplicate is the Appeal Brief in this application, with respect to the Notice of Appeal filed on

RECEIVED
2001 FEB 26 PM 2:46
BOARD OF PATENT APPEALS AND INTERFERENCES

The fee for filing this Appeal Brief is: **$310.00**

☒ A check in the amount of the fee is enclosed.

☐ The Commissioner has already been authorized to charge fees in this application to a Deposit Account. A duplicate copy of this sheet is enclosed.

☒ The Commissioner is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. **20-1504**
A duplicate copy of this sheet is enclosed.

Signature

Dated: 2/23/01

Timothy N. Trop, Reg. 28,994
TROP, PRUNER & HU, P.C.
8554 Katy Fwy, Ste 100
Houston, TX 77024
713/468-8880 [Phone]
713/468-8883 [Fax]

I certify that this document and fee is being deposited on 2-23-01 with the U.S. Postal Service as first class mail under 37 C.F.R. 1.8 and is addressed to the Assistant Commissioner for Patents, Washington, D.C. 20231.

Signature of Person Mailing Correspondence

Azilda Marcel Authement

Typed or Printed Name of Person Mailing Correspondence

cc:

#14

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	JOHN W. MERRILL	§	Group Art Unit:	2743
		§		
Serial No.:	09/115,359	§		
		§	Examiner:	R. Sax
Filed:	July 14, 1998	§		
		§		
For:	AUTOMATIC SPEECH RECOGNITION	§	Atty. Dkt. No.:	INTL-0038-US (P5634)
		§		

Board of Patent Appeals & Interferences
Commissioner for Patents
Washington, D.C. 20231

APPEAL BRIEF

Sir:

Applicant respectfully appeals from the final rejection mailed October 11, 2000.

I. REAL PARTY IN INTEREST

The real party in interest is the assignee Intel Corporation.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 14, 15 and 21 through 32 are rejected. Each rejection is appealed.

IV. STATUS OF AMENDMENTS

All amendments were entered. It is believed that no response was received to the reply to final rejection because the Examiner was on sick leave.

Date of Deposit: 2-23-01
I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage on the date indicated above and is addressed to the Board of Patent Appeals & Interferences, Commissioner for Patents, Washington, DC 20231.
Azilda Marcel Authement
Azilda Marcel Authement

V. SUMMARY OF THE INVENTION

Referring to Fig. 1, a speech recognition system 11 involves an application software program 10 which needs to respond to spoken commands. For example, the application 10 may be implemented through various graphical user interfaces or windows in association with the Windows® operating system. Those windows may call for user selection of various tasks or control inputs. The application 10 may respond either to spoken commands or tactile inputs. Tactile inputs could include pushing a keyboard key, touching a display screen, or mouse clicking on a visual interface.

The application 10 communicates with a server 12. In an object oriented programming language, the server 12 could be a container. In the illustrated embodiment, the server 12 communicates with the control 14 which could be an object or an ActiveX control, for example. The control 14 also communicates directly with the application 10.

The server 12 can call the speech recognition engine 16. At the same time, a button driver 18 can provide inputs to the server 12 and the control 14. Thus, in some embodiments, the control 14 can receive either spoken or tactile inputs (from the button driver 18) and acts in response to each type of input in essentially the same way.

Referring to Fig. 2, a program for recognizing speech may involve beginning an application (block 90) that needs speech recognition services. The speech engine is provided with a vocabulary of command sets for an active screen or task, as indicated in block 92. The command sets could be the vocabulary for each of the various applications that are implemented by the particular computer system. The command set for the current application that is currently running is communicated to the server 12 or control 14 (block 94). Next, the speech is

recognized and appropriate actions are taken, as indicated in block 96. See Specification at page 3, line 10 through page 4, line 29.

Another implementation, shown in Fig. 3, also begins with starting an application, as indicated in block 98. Speech units that need to be decoded are associated with identifiers (block 100). The identifiers may then be associated with a particular action to be taken in the application in response to the spoken command (block 102). Next, the flow determines the identifier for a particular spoken speech unit (block 104). The identifier is provided to a software object such as the control 14, as indicated in block 106. An event is fired when the object receives the command, as shown in block 108. The event may be fired by the object whether the command is a result of a spoken command or a tactilely generated command.

Referring to Fig. 4, the application 10 passes a grammar table to the server 12 (block 20). In particular, the application initializes the grammar with speech identifiers associated with each spoken command used in the application. These commands make up all of the command sets for a given engine. The grammar is a set of commands that may include alternative phrases. For example, a simple grammar could be (start/begin)(navigator). This grammar would respond to the spoken commands "start navigator" and "begin navigator".

The speech recognition engine 16 can operate on phonemes or with discrete terms. Thus, the application provides the particular command set (which is a subset of the engine's available commands) with the active application. This facilitates speech recognition because the speech recognition engine can be advised of the particular words (command set) that are likely to be used in the particular application that is running. Thus, the speech recognition engine only needs to match the spoken words with a smaller sub-vocabulary. For example, if the navigator

function was operating, only the command set of words associated with that application need be decoded. See Specification at page 4, line 30 through page 6, line 4.

In response, the server 12 initializes the speech engine 16 (block 22). The server 12 has a phrase and identifier table 36 as indicated in Fig. 1. The application 10 also sends the speech identifiers associated with given spoken commands to the control 14 or server 12 (block 24). When the control 14 is activated in the container or server, the control may call the OnControlInfoChanged method in the IOleControlSite interface, in an embodiment using ActiveX controls. This provides for transfer of information from the control 14 to the server 12 (block 26). The server in turn may call the GetControlInfo method from the IOleControl interface which allows communications from the server or container 12 to the control 14 (block 28).

The server uses the GetControlInfo method in the IOleControl interface and the OnMnemonic method in IOleControl to request identifiers from the control. The control may provide this information through IOleControlSite interface and the OnControlInfoChanged method, using ActiveX technology for example.

The server 12 enables the speech engine 16 (block 30), for any commands that are active, from the server's table 36. The server uses the table 36 from the application to provide focus in particular applications. The control provides an effect comparable to that of an accelerator key. Namely, it provides a function that can be invoked from any window or frame reference. The application provides the speech identifiers and associates the identifiers with an action by the control. See Specification at page 6, line 5 through page 7, line 2.

The server knows which vocabulary to use based on what task is running currently. In a system using windows this would correspond to the active screen. Thus, if the navigator is running, the server knows what the sub-vocabulary is that must be recognized by the speech engine.

When the server receives a speech message, it calls the speech API in the engine 16. When a phrase is detected, the engine provides the phrase to the server for example, as a text message. The container does a table look-up (block 32). On a match between the phrase and the identifier, the server 12 may call the OnMnemonic method of the IOleControl interface, passing the identifier to the control. The control follows its preprogrammed rules and implements the corresponding action (block 34). The control may handle the message internally or send an event to the server.

As a simple example, a given screen may include two buttons, “ok” and “delete”. When the application comes up it sends the grammar for this screen to the server. For example, the grammar for “ok” might include “ok”, “right” and “correct”.

The application then associates “ok” with an identifier which corresponds to a particular control and does the same thing with “delete”. The identifier is simply a pointer or handle that is unique, within the application, to the particular command. The table 36 then includes the phrases “ok” and “delete”, an identifier for each phrase and an identifier for the control that handles the command.

When a control is instantiated, the application provides it with its identifier. The control is preprogrammed with the action it will take when the server advises the control that its identifier has been called. See Specification at page 7, line 3 through page 8, line 5.

When a speaker uses a word, the speech engine sends the word to the server. The server checks the phases in its table 36 to see if the word is in its active list. In the simple example, if the word sent by the speech engine is not "ok" or "delete," it is discarded. This would indicate a speech engine error. If there is a match between the word and the active vocabulary, the server sends the appropriate control identifier to the appropriate control, which then acts according to its programmed instructions.

A phoneme based speech engine with a large vocabulary can be used with high reliability because the engine is focused on a limited vocabulary at any given time. Advantageously this limited vocabulary may be less than 20 words in the table 36 at any given instance.

This frees the application from having to keep track of the active vocabulary. The server can tell the server which words to watch for at a given instance based on the active task's vocabulary.

There may also be a global vocabulary that is always available regardless of the active screen. For example, there may be a "Jump" command to switch screens or an "Off" command to terminate the active task.

Advantageously, the existing mnemonics or "hot keys" available in Microsoft Windows® may be used to implement speech recognition. For example, the OnMnemonic method may be given the new function of passing information from the server to the control corresponding to a spoken command. See Specification at page 8, line 4 through page 8, line 30.

With embodiments of the present invention, an effect comparable to that of an accelerator key is provided. It gives a focus to the command with reference to a particular application. Therefore, speech can be used to focus between two operating tasks. For example, as shown in

Fig. 5, if two windows A and B are open at the same time on the screen 76, the command that is spoken can be recognized as being associated with one of the two active task windows or frames. Referring to Fig. 6, after a command is recognized (block 78), the application provides information about what is the primary, currently operating task and the speech may be associated with that particular task to provide focus (block 80). An input is then provided to one of the tasks (and not the other), as indicated at block 82. In this way, the speech recognition is accomplished in a way which is effectively invisible to the application. To the application, it seems as though the operating system is effectively doing the speech recognition function. Synchronization is largely unnecessary.

The message which is passed to the ActiveX control from the container can include a field which allows the application to know if the command was speech generated. This may be useful, for example, when it is desired to given a spoken response to a spoken command. Otherwise, the application is basically oblivious to whether or not the command was speech generated or tactilely generated. See Specification at page 9, line 6 through page 10, line 1.

While the application loads the identifiers into the ActiveX controls (when they are instantiated), the controls and the container handle all of the speech recognition for the command words. The control and its container are responsible for managing when the words are valid and for sending appropriate messages to the application. Thus, the container or server does all the communication with the speech recognition API. The container may communicate with the ActiveX controls by standard interfaces such as IOleControl. As a result, the number of state errors that would otherwise occur if the application were forced to handle the speech recognition itself. See Specification at page 10, line 2 through page 10, line 13.

VI. ISSUES

A. Is The § 102 Rejection Of Claims 14, 15, and 25-32 Proper?

B. Is The § 112 Rejection of Claims 26 and 30 Proper?

C. Is The § 103 Rejection of Claims 21-24 Proper?

VII. GROUPING OF THE CLAIMS

The claims should not be grouped.

VIII. ARGUMENT

A. Is The § 102 Rejection Of Claims 14, 15, and 25 –32 Proper?

Claims 14, 15 and 25-32 are rejected as being anticipated by Trower et al.

Claim 14 calls for providing a software object that receives spoken and non-spoken command information. An event is fired when the object receives command information.

Trower does not teach an object that receives both spoken and non-spoken command information.

In the final rejection at paragraph 4, the Examiner misquotes claim 14. The misquote is material because the Examiner asserts that the claim states that the object must respond to either spoken <u>or</u> non-spoken commands. This is not the case. The claimed object responds to both. It is clear that the reference may respond to one or the other but not both.

This is made explicitly clear by the very passage relied upon by the Examiner. The commands object in the reference has a voice property which species that the object may respond to either spoken or non-spoken commands, but not both. See column 27, lines 25-26. By the Examiner's own reasoning, the § 102 rejection should be withdrawn.

The same rationale applies to claim 15.

Similarly, claims 25-30, dependent thereon, should also be in condition for allowance.

The same language is included in claims 31 and 32 and therefore these claims also distinguish over the art of record.

B. Is The § 112 Rejection Of Claims 26 and 30 Proper?

Reconsideration of the § 112 rejections of claims 26 and 30 is requested. Claims 26 and 30 were objected to on the grounds that there is no antecedent basis for "object" in each claim. However, claim 26 calls for "instantiating an object". This provides the antecedent basis for the next instance of object namely, "communicating the identifier to the object". The reference to object in claim 30 goes back to claim 29 and from there to claim 15. There is antecedent basis for "object" in claim 15.

C. Is The § 103 Rejection of Claims 21-24 Proper?

Claims 21-24 are rejected under § 103 based on the Hashimoto reference alone. The Examiner contends that "it would have been obvious to infer that the same action produced in response to any standard input, whether inputted to the application by speech I/O or other I/O has the same identifier which would have been inferred to have been the label for activating the object code executed in response to the spoken or non-spoken command input." See office action page 9.

It is respectfully submitted that this is not a proper obviousness rejection. To make out a *prima facie* obviousness rejection, the examiner must point out the deficiency in the prior art reference and provide a rationale to modify the reference period. Further, the Examiner must point out something in the prior art that teaches the missing element. Here the Examiner merely presumes that the same identifier must be used for both actions. However, there is no rationale to modify the reference to make this conclusion except through the use of hindsight reasoning. Moreover, the Examiner must point to something in the prior art that teaches the missing element and provides the rationale to combine. Here the Examiner has failed to do so and therefore a *prima facie* rejection is not made out.

Absent a *prima facie* rejection, the applicant has no duty to respond. The failure to provide a *prima facie* rejection makes the rejection improper on its face.

It is respectfully suggested that perhaps the Examiner's argument is not one of obviousness but rather one of inherency. In other words, what the Examiner is arguing is that the Examiner believes that inherently the reference must operate in the claimed fashion. However, this too would be mistaken since there is no reason that the same identifier must be provided for both spoken and non-spoken commands. Clearly, different identifiers could be utilized in each case and the two commands could be handled completely independently.

Not only is this possible, but this is explicitly what Hashimoto says that he does. At column 18, lines 53-60, Hashimoto's states that in the configuration of Figure 17 (relied upon by the Examiner in the office action), "the speech recognition system 3 and the window system 4 are operating <u>independently from each other</u>". [Emphasis added].

Given Hashimoto's description of his own system, there is no reason to believe that the same identifier is utilized to handle either spoken or non-spoken commands. Instead, it is explicitly clear that Hashimoto fails to teach the use of the same identifier. Therefore, the missing element can not be inherently taught by Hashimoto. To the contrary, Hasimoto explicitly teaches away from the claimed invention.

Hashimoto can not render the claimed invention obvious. Instead, Hashimoto conventionally teaches away from the claimed solution by suggesting two independent systems for handling spoken and non-spoken commands. Absent any rationale to modify Hashimoto <u>against his own teaching</u>, or anything in the prior art that would teach the modification of Hashimoto to achieve the claimed invention, the rejection should be reconsidered.

IX. CONCLUSION

Since the rejections of the claims are baseless, they should be reversed.

Respectfully submitted,

Date: 2/23/01

Timothy N. Trop, Reg. No. 28,994
TROP, PRUNER & HU, P.C.
8554 Katy Fwy, Ste 100
Houston, TX 77024-1805
713/468-8880 [Phone]
713/468-8883 [Facsimile]

APPENDIX OF CLAIMS

The claims on appeal are:

14. A method for responding to user inputs to a computer comprising:

providing a software object that receives spoken and non-spoken command information; and

firing an event when an object receives command information.

15. An article comprising a machine readable storage medium that stores instructions that cause a computer to:

provide a software object that receives spoken and non-spoken command information; and

fire an event when the object receives command information.

21. A method comprising:

associating a spoken and a non-spoken command with the same identifier;

associating the identifier with an action to be taken in response to either the spoken or non-spoken command;

determining the identifier for a spoken or non-spoken command; and

providing the identifier to a software object.

22. The method of claim 21 including instantiating an object in a container and communicating the identifier to the object when a particular command is received.

23. The method of claim 22 including communicating information about a first spoken command to the container, checking an active vocabulary list in the container to determine if the first spoken command is one used in an active task, and if the first spoken command is one used in an active task, transferring the identifier for the spoken command to the object.

24. The method of claim 21 including using an OnMnemonic method to communicate between the container and the object.

25. The method of claim 14 including:

associating speech commands with identifiers;

associating the identifiers with actions to be taken in response to each speech command;

determining the identifier for a spoken speech command; and

providing an identifier to a software agent.

26. The method of claim 25 including instantiating an object in a container and communicating the identifier to the object when a particular a particular speech command is spoken.

27. The method of claim 26 including communicating information about a first speech command to the container, checking an active vocabulary list on the container to

determine if the first speech command is one used in an active task, and if the speech command is one used in an active task, transmitting identifier for the speech command to the object.

28. The method of claim 26 including using an OnMnemonic method to communicate between the container and the object.

29. The article of claim 15 further storing instructions that cause a computer to associate spoken commands with an identifier, associate the identifier with actions to be taken in response to each command, determine the identifier for a spoken command, and provide the identifier to a software agent.

30. The article of claim 29 further storing instructions that cause a computer system to communicate information about a first command to the controller, check an active vocabulary list in the container to determine if the first command is one used in an active task, and if the first command is one used in an active task, transfer the identifier for that command to the object.

31. A method for responding to user inputs to a computer comprising:

providing a software object that responds to spoken and non-spoken command information to implement the same command; and

firing the same event when said object receives spoken and non-spoken command information.

32. An article comprising a medium storing instructions that enable a computer to:

provide a software object that responds to spoken and non-spoken command information to implement the same command; and

fire the same event when said object receives spoken and non-spoken command information.